AD-A233 075

ITATION PAGE

Form Approved
OMB No. 0704-0188

d to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, viewing the collection of information. Send comments regarding this burden estimate or any other aspect of burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson ffice of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. Agency Use Only (Leave blank).	2. Report Date. 1990	3. Report Type and Dates Covered. Proceedings

4. Title and Subtitle. Managing Large Scale Knowledge Bases	5. Funding Numbers. Program Element No. 256020N; Project No. 00101; Task No. 101; Accession No. DN257009
6. Author(s). E. Wade Nation	
7. Performing Organization Name(s) and Address(es). Naval Oceanographic and Atmospheric Research Laboratory Stennis Space Center, MS 39529-5004	8. Performing Organization Report Number. PR 89:078:252
9. Sponsoring/Monitoring Agency Name(s) and Address(es). Naval Sea Systems Command Washington, D.C. 20362-5101	10. Sponsoring/Monitoring Agency Report Number. PR 89:078:252
11. Supplementary Notes. IEEE	DTIC ELECTE
12a. Distribution/Availability Statement. Approved for public release; distribution is unlimited.	12b. Distribution Code. S APR 01 1991 D G

13. Abstract (Maximum 200 words).

The development of large scale expert systems through traditional methods will in most cases present performance problems. The single large knowledge base created with off-the-shelf development tools is not the ideal structure for expert systems based primarily on theoretical knowledge. The research results presented in this paper illustrate a segregated knowledge base structure designed to increase performance in large scale highly non-empirical knowledge bases.

14. Subject Terms. (U) Expert Systems; (U) ASW Scenarios; (U) Artificial Intelligence; (U) Maintenance Advisors	15. Number of Pages. 3
	16. Price Code.

17. Security Classification of Report. Unclassified	18. Security Classification of This Page. Unclassified	19. Security Classification of Abstract. Unclassified	20. Limitation of Abstract. SAR

NSN 7540-01-280-5500

91 3 12 009

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. Z39-18
298-102

IEEE

SOUTHEASTCON '90

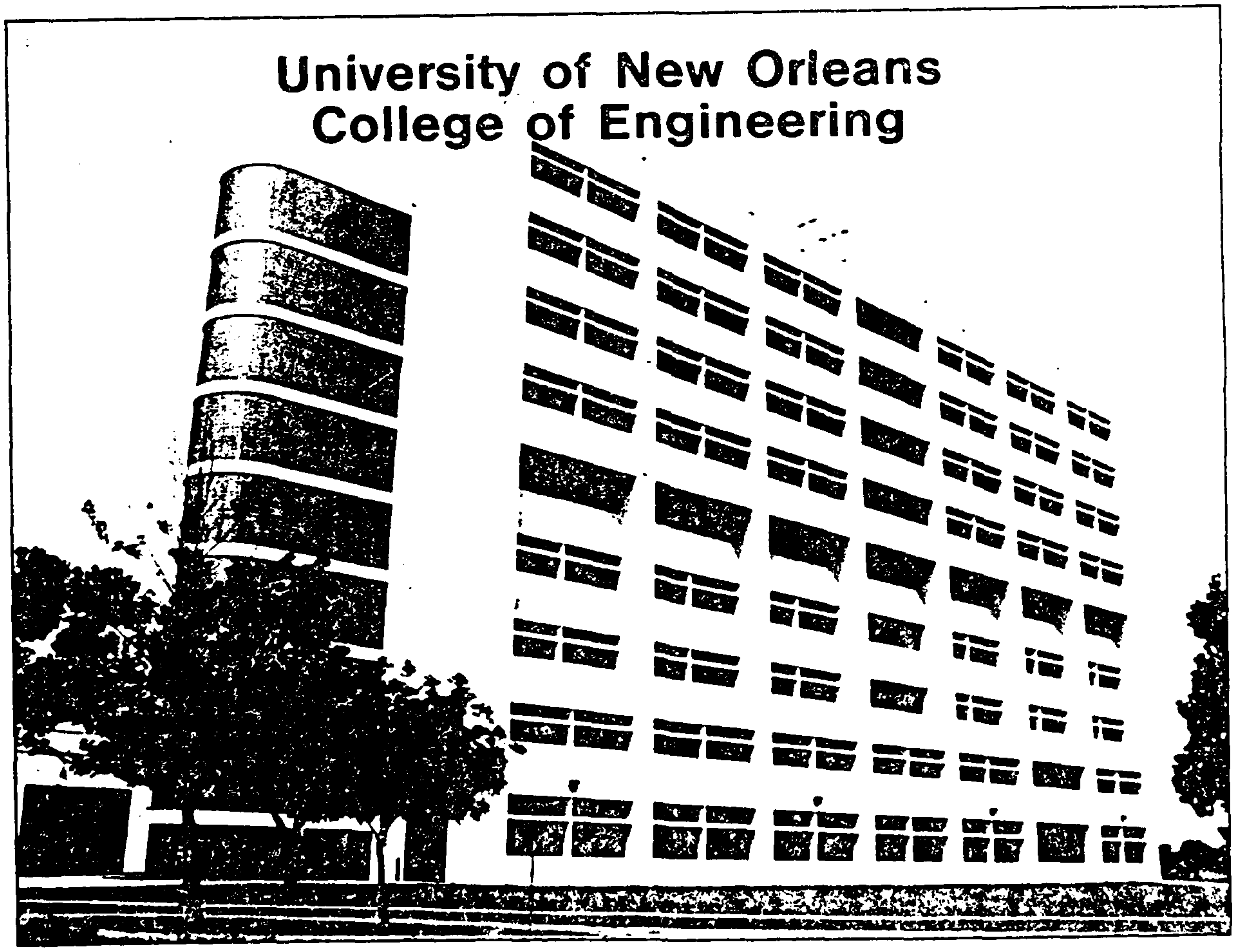
PROCEEDINGS

TECHNOLOGIES ODAY AND TOMORROW

Conference and Exhibit

IEEE

Volume 1 of 3



Hosted By

IEEE New Orleans Section
IEEE Region 3
IEEE Area 6

University of New Orleans
Tulane University
New Orleans, Louisiana

April 1-4, 1990
Conference and Exhibit

Proceedings
Compliments of
South Central Bell
90CH2883-7

Managing Large Scale Knowledge Bases

E. Wade Nation
Advanced Technology Branch, Code 252
Naval Oceanographic and
Atmospheric Research Laboratory
Stennis Space Center, MS 39529-5004

Abstract

The development of large scale expert systems through traditional methods will in most cases present performance problems. The single large knowledge base created with off- the-shelf development tools is not the ideal structure for expert systems based primarily on theoretical knowledge. The research results presented in this paper illustrate a segregated knowledge base structure designed to increase performance in large scale highly non-empirical knowledge bases.

Introduction

Applying Artificial Intelligence (AI), specifically Expert Systems (ES), to real world problems can at times be a very significant undertaking, even under the strict confines of traditional ES development. Many potential applications suitable to ES techniques are not implemented due to the inability to identify domain experts, developmental time constraints or poor performance of prototypes. This case study includes all three implementation obstacles and the identified research results used to minimize their impact on the final product. The thesis of this paper is that large scale expert systems can be developed in the face of development time constraints, poorly performing prototypes, and the lack of true domain experts. Smaller ES generally do not suffer from these obstacles, but in most cases large scale projects are subjected to at least one and many times all three of these limitations.

Traditional ES development utilizing off-the-shelf shells lend themselves to a single contiguous knowledge base. The economy of such a structure is lost for large scale applications by the inability of the inference engine to traverse large contiguous knowledge bases efficiently. The human mind, when presented with a problem in a diverse environment, manages very well to isolate the problem from extraneous stimulus. The mind can somehow sift through and ignore a large portion of its knowledge about a subject and quickly move to the problem area. The inference engine on the other hand does not perform as well on large knowledge bases. Large scale ES tend to be created primarily with theoretical knowledge or ready-made knowledge. It has been our experience that the use of empirical knowledge or experience tends to decrease as the size of the subject system increases because domain experts are generally limited and the subject matter involved is very broad. The theoretical knowledge is usually obtained through operation and maintenance manuals, manufacturers specifications, and other literature. Knowledge bases created primarily on theoretical knowledge suffer performance problems in general due to the inability of the inference engine to quickly move to the area of the knowledge base related to the issue at hand. The inference engine must traverse the large contiguous knowledge base in an almost procedural fashion, and unlike the human mind, it is subject to processing a large degree of knowledge not related to the specific problem.

Through applied research, one method to more nearly approximate the results obtained by the mind has been developed. The implementation of a hierarchical knowledge base design approach has reduced the impact of large amounts of theoretical knowledge on system performance. The implementation of a hierarchical knowledge base also offers the ability to divide the knowledge engineering function along segregated knowledge base boundaries. Through this division of the knowledge base, the overall development time should be reduced in direct proportion to the number of knowledge engineers assigned to the task. The knowledge engineers utilized are different from the classical image, since the knowledge is mostly theoretical, in that the majority should have a background in the problem discipline (i.e. electrical, mechanical, chemical), rather than posses high knowledge acquisition skills. It is much easier and less expensive to find a good junior engineer or technician to independently encode knowledge from a blueprint or schematic rather than try to find derived knowledge through the classical knowledge engineering route.

Case Study

The case study presented is for a maintenance advisor for U.S. Navy sonar systems. The size of the sonar system is very large, consisting of several decks of cabinets, each standing about six feet tall, each using about nine square feet of floor space and filled with racks of printed circuit boards. Our goal was to demonstrate the applicability of ES techniques to assist sonar technicians in the isolation of malfunctions to the component level. We were careful to select our candidate by using the traditional ES implementation guidelines. The candidate area was well bounded, static, and of sufficient technical challenge to be an asset to sonar maintenance technicians.

Phase One development included the identification of the development system, including inference engine and hardware. The Fault Isolation System (FIS), developed by the Navy Center for Applied Research In AI, was used as the inference engine. While there were certain financial considerations exercised in the selection of FIS, the system in general seemed well suited to our maintenance advisor. FIS had been developed primarily as a development tool for aircraft systems maintenance. FIS is LISP based and uses frame based knowledge representation of the system. FIS uses statistics and probabilities to isolate the problem area and does not contain rules. FIS is designed primarily for theoretical knowledge through the frame representation, while derived knowledge can be included by adjusting initial fault probabilities. FIS was installed on a SUN 3 and frame representation was conducted by a senior electrical engineer.

About 3,000 frames later, the circuitry in a single cabinet had been encoded into the knowledge base. The first test of the system was more than disappointing. The performance was very slow with system prompts occurring at intervals greater than thirty seconds. The system could not hold the interest of the sonar technician, and the ability to expand the system was a long way removed. Obviously, we could not present this product as a prototypical application of expert systems technology to real world maintenance issues. We had two options, to find more computer power or to redesign the system. We felt pretty comfortable that a Cray Supercomputer would present our system in a much better light, but the reality that most ships do not possess computing power on the order of a Cray, limited us in that area. Our second option, to redesign the system, offered the most realistic path to follow.

Phase Two of the maintenance advisor project was primarily devoted to the redesign effort. We started at the beginning with a more thorough evaluation of our development tools and identified that FIS response time was being consumed by the large amount of knowledge it had to traverse with each inference. The demonstrations we viewed in Phase One evaluation were conducted with small knowledge bases and left out a few limiting details of the system we overlooked. FIS had been written to assume that all the knowledge about a system was available to the inference engine and FIS did not have the capability to gracefully exchange knowledge bases during a maintenance effort. Also, FIS did not have multiple fault detection or diagnosis capability, when a fault was identified the maintenance activity terminated.

Our prototype application may have allowed us to build a single large contiguous knowledge base, but a sonar system- wide application made a single knowledge base approach not very desirable from a software maintenance viewpoint. The inability of FIS to perform multiple fault detection did not parallel our concept of the real world. Obviously, had we known these facts in Phase One and we definitely should have devoted more time researching our tools, our inference engine may have been very different, but by this time we had a significant investment in FIS. With the help of the Navy Center for Applied Research in AI and the Naval Research Laboratory, we modified FIS to be more agreeable to large real world problems. The modified version of FIS became the Fault Isolation Layer (FIL). This review of our development tools presented in a roundabout way the solution to our performance issue. While we modified FIS to allow knowledge base substitution for any expanded sonar maintenance advisor applications, it dawned on us that by breaking our problem down into smaller pieces and substituting a number of smaller, more efficient knowledge bases that the performance would be increased. As simple as it may sound, the challenges were as great as the idea was simple. Analog electronic circuits are highly integrated; power and timing are generally produced in one area but utilized throughout the circuit. Feedback and other control signals can be created downstream of the test point and yet have a significant impact on their own value. The division of the knowledge base while not difficult did require some forethought.

Research Results

The remainder of this paper is devoted to the presentation of the methodology for handling the knowledge base substitution. The actual knowledge base boundaries within the circuity will differ with each application and are therefore not illustrated here. The ability to reduce development time through the utilization of additional knowledge engineers will not be proven in this paper but is presented as a logical fact. The final system cost will not necessarily decrease with development time due to the added man-power and generally consistent with total man-hours used to develop the system. With all those caveats stated, the file structures and the communication strategies are the principle results of the research.

The maintenance advisor was developed in a hierarchical manner, with master control performed by a Local Area Expert (LAE). The LAE represents a specific expert system for the maintenance effort, and in our case consisting of high level knowledge for a sonar electrical cabinet. The slave function is performed by the Fault Isolation Layer (FIL). FIL is a generic expert system that uses the knowledge passed by LAE to generate test sequences. The knowledge base passed to FIL is considered the Unit Under Test (UUT).

A typical troubleshooting scenario would consist of the following flow of knowledge. When the system is started the user is accessing the LAE, the LAE prompts certain questions in an effort to identify the proper UUT to send to FIL. When LAE has determined the most logical UUT to begin trouble shooting, the appropriate knowledge base is passed to FIL. FIL performs detailed troubleshooting in efforts to identify the fault and controls the user interface. If a fault is located or if no fault is found within the UUT, FIL updates the LAE control files and LAE proceeds to identify the next most appropriate UUT for testing. If faults remain, control is again given to FIL and the process continues until no faults remain.

There are three files which the LAE accesses with two of these files being used to interface with FIL. The Valids Data File and the Invalids Data File are used as by LAE and FIL to communicate. The Tests Data File is used locally by LAE to store all the results of test performed by LAE and FIL.

The Tests Data File is a binary data file and it contains a header followed by a variable number of data records. The header record contains the count variable of the number of test and a status variable. The current status of the LAE-FIL operation cycle is indicated by the status variable and may be one of the following values: "NEW", "FIL", "LAE", or "BAD". If the value is "NEW", the LAE performs its local startup test. If the value is "FIL", LAE begins by loading test results from the Valids and the Invalids files into the Test Data File or if the status is "LAE" or "BAD" then LAE performs a recovery operation.

The variable record portion of the Tests Data File consist of the UUT number for the test, the test terminal identifier, the test parameter, and the qualitative value for the test. A single test can affect several UUT so a simple code is used for the UUT number. A UUT number field is binary with a high bit used to indicate that the test is included in a particular UUT. A sixteen bit field can represent a test being included in any combination of sixteen distinct UUTs.

The terminal name represents a printed circuit board terminal. The test parameter defines the operational parameter specific to the terminal. The qualitative value for the test is the value assigned to the tested terminal parameter. FIL defines various qualitative values for different terminal parameters, such as "GOOD", "HIGH", and "LOW". If a terminal parameter has been invalidated, "INVALID" will be stored in the qualitative field. This "INVALID" value is assigned to tests downstream of replaced components.

The Invalids File is a binary data file used to communicate between LAE and FIL and it consists of a header record and a variable number of data records. The header contains two variables, a count of the number of invalidated test in the file and the current UUT. Once LAE determines that a UUTneeds to be tested, LAE loads the UUT variable with the code number that represents the UUT. The code representation is identical to the Tests Data File, except that only a single UUT is indicated by one set UUT bit flag and this number informs FIL about which UUT to test. Once FIL completes its testing process, it returns the UUT number to process next test or ends, depending on the results of FIL's testing.

Each Invalids Data File data record contains the names of a terminal parameter which have been invalidated through a replaced component procedure in FIL. When LAE sends an Invalids Data File to FIL, all previously invalidated terminal parameters associated with a UUT will be loaded into the file. Upon return, FIL will send a new list of terminal parameters which should be invalidated by LAE.

The Valids Data File is an ASCII LISP data file used by LAE and FIL to transfer test information. When LAE determines that FIL needs to test a UUT, it loads the qualitative values of all valid terminal parameters in the Tests Data File, which are associated with the UUT to be tested, into the Valids Data File. This data is written in a format which the LISP based FIL shell can easily interpret. Once FIL has completed testing, it returns a new list of terminal parameter qualitative values to save in LAE's Test Data File data base.

The format of the Valids Data File is best shown by example, other than by "Lots of Irritating Stacked Parenthesis (LISP)". Each record contains a terminal name, a parameter name, the string "S1", the string "SS1", and a qualitative value enclosed in parenthesis. All records in the file are also enclosed an additional set of parenthesis. "S1" and "SS1" are blank fields for later implementation if needed.

Example: ((term1 param1 S1 SS1 qualval1)(term2 param2 S1 SS1 qualval2)(term3 param3 S1 SS1 qualval3))

With these three files total control and efficient communication exist between LAE and FIL. The knowledge bases are simply separate files that are loaded by FIL under direction of these files. At the completion of testing under FIL, if components have been replaced then a global invalidation of downstream test is performed to ensure integrity of the maintenance effort and the process of knowledge base substitution is continued until no faults remain. The performance increase experienced with this structure was outstanding. While Phase 1 prompted on the order of thirty seconds; the Phase 2 implementation issued next test prompts on an average of two seconds. The reduction of the size of the knowledge base FIL needed to traverse was greatly reduced and the additional time required to load the next UUT was insignificant, since it generally required the technician to prepare for the next test area while the UUT was loading.

This hierarchical approach and the segregation of the knowledge base lends itself to large scale projects. Through this structure, the response time can be adjusted to within ranges acceptable to the technician. The upper limit to the scope of a maintenance advisor utilizing this structure is unbounded. An additional layer, say a Global Expert (GE), could be used to transfer control to the proper LAE on very large systems.

Acknowledgements

This paper is approved for public release; distribution is unlimited. The research results presented in this paper were contributed as number PR-89:078:252 and sponsored by the Surface Ship ASW Combat System Project through PMS411 under the Naval Sea Systems Command, program element 25620N.

E. WADE NATION

As Principal Investigator and Group Leader of the Intelligent Systems Group within the Advanced Technology Branch of NOARL, Wade Nation conducts research on Artificial Intelligence and Expert Systems to assist U.S. Navy operations. His primary research areas are Expert Maintenance Advisor and Automated ASW training through computer-controlled gaming exercises. He received his M.S. in Computer Science from the University of Southern Mississippi.